JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING 8, 340–348 (1990)

Squashed Embedding of E-R Schemas in Hypercubes

CHAITANYA K. BARU* AND PIYUSH GOEL

Advanced Computer Architecture Laboratory, Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, Michigan 48109

We have been investigating an approach to parallel database processing based on treating Entity–Relationship (E-R) schema graphs as dataflow graphs. A prerequisite is to find appropriate embeddings of the schema graphs into a processor graph, in this case a hypercube. This paper studies a class of adjacency preserving embeddings that map a node in the schema graph into a subcube (*relaxed squashed* or *RS* embeddings) or into adjacent subcubes (relaxed *extended* squashed or *RES* embeddings) of a hypercube. The mapping algorithm is motivated by the technique used for state assignment in asynchronous sequential machines. In general, the dimension of the cube required for squashed embedding of a graph is called the *weak cubical dimension* or *WCD* of the graph. The RES embedding provides an RES-WCD of $O(\lceil \log_2 n \rceil)$ for a completely connected graph, K_n, and RS embedding provides an RS-WCD of $O(\lceil \log_2 n \rceil + \lceil \log_2 m \rceil)$ for a completely connected bigraph, $K_{m,n}$. Typical E-R graphs are incompletely connected bigraphs. An algorithm for embedding incomplete bigraphs is presented.

1. INTRODUCTION

This paper presents one aspect of a novel approach to database processing which is based on treating schema graphs as dataflow graphs and then mapping such graphs onto a set of processors to support parallel database processing. Entity–Relationship, or E-R, schema graphs [10] and hypercube multicomputers are considered here. Adjacent nodes (object classes) in the schema graph are mapped to adjacent subcubes in the hypercube. This type of mapping is referred to as relaxed squashed embedding.

Semantic data models were introduced as aids to the easy modeling of data and the various interrelationships among them. The E-R model is a well-known example of such a data model. Many other semantic data models have also been proposed, including variants of the E-R model [15, 29] and other models such as SDM [19], SAM* [27], NIAM [30], and IFO [1]. Semantic data models typically support constructs such as aggregation, generalization, and classification. Most models are graph-oriented; i.e., the various interrelationships among data objects are represented via a graph employing several different symbols to identify different object and relationship types. This graph is referred to as the *logical schema graph.* Figure 1 shows a logical schema graph for a simple database using a variant of the E-R model, viz. the Entity–Category–Relationship (ECR) model [15].

An important concept in the work presented here is the use of the logical schema graph as the physical representational structure; i.e., the data are directly stored in the computer in the form indicated by the schema graph. Each node in the graph represents a collection of objects or relationships among objects. These are stored directly as files, either flat or structured. Several points should be noted here. First, user queries can always be represented as subgraphs of the schema graph, with appropriate selection predicates at each node. Second, a dataflow processing approach can be employed by treating the query graph as a dataflow graph. Third, the dataflow processing is made more efficient if each node of the graph resides on a different processor with processors being connected by point-to-point links as indicated by the edges in the graph. This last observation provides the motivation for the work reported in this paper.

The rest of this paper is organized as follows. Section 2 provides a brief survey of related areas in database processing and graph embeddings. Section 3 provides upper bounds on the cube dimensions for squashed embeddings of completely connected graphs and complete bigraphs. Also, an algorithm for mapping incompletely connected bigraphs is provided. An example of an E-R database schema is given in Section 4, along with a discussion of the advantages and disadvantages of the proposed embedding scheme for parallel database processing. Finally, Section 5 provides conclusions.

2. SURVEY OF RELATED AREAS

2.1. Related Database Work

Direct encoding of data in the form represented by the E-R schema and the use of a dataflow approach to process these data are both techniques that were used in the Active Graph Model (AGM) machine [6–8]. However, the AGM approach is completely different from the one proposed

* Partially supported by Grant IRI-8710855 from the National Science Foundation.

0743-7315/90 $3.00

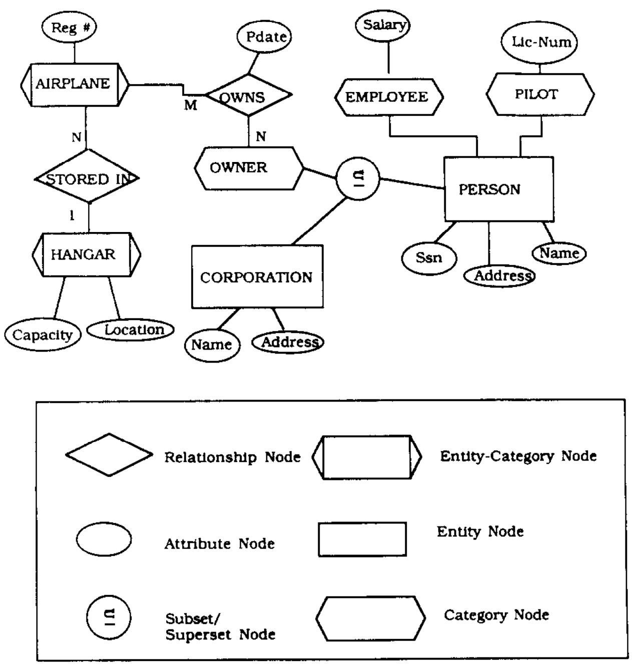


FIG. 1. An ECR logical schema graph for a hypothetical airport database.

here since in AGM each object class (entity or relationship) *occurrence* is hashed onto a processor. The processors in AGM are connected in the form of a 2-D torus and each processor is a special, dataflow processor. Each occurrence that is selected by the query is routed within the 2-D torus to the location of its related entity (or entities). Each occurrence corresponds to a token in the typical dataflow method of processing. In comparison, the approach proposed here employs *macro* dataflow processing, where entire sets of selected entity occurrences, rather than individual tokens, are transmitted between nodes. Also, due to the graph mapping technique employed, only the processors containing the data referenced by the query are involved in processing the query, rather than all the processors in the system.

Some issues in relational database processing on hypercube multicomputers have been addressed in our past research [3]. A multicomputer system is one in which each node is an independent computer. As an extension to this research, several parallel algorithms for the relational join operation were implemented on a 64-node NCUBE [4] and parallel query execution strategies were studied via simulations [26]. Further implementations will be carried out on a transputer-based, parallel computer that we have acquired for this project. Another research project that has considered the hypercube interconnection scheme for parallel database processing is [9].

2.2. *Graph Embedding in Hypercubes*

A variety of graph embeddings in hypercubes have been studied. A survey of some of this work is available in [25]. A typical approach is to view the hypercube multicomputer as a processor graph and one can then study the problem of embedding *task graphs* into this processor graph. In database applications, the *query graphs* are the task graphs and they are subgraphs of the logical schema graph. The nodes in the query graph represent the objects (entities and/or relationships) and the edges represent the joins (or similar operations) that need to be performed between the objects. In order to keep the communication overhead low, adjacent nodes of the logical schema graph should be assigned to adjacent nodes in Q_n, if possible.

Several classes of embedding have been proposed to embed a given graph, G, into a hypercube, Q_d [20]. An embedding of a graph $G = (V, E)$ into a $Q_d = (V', E')$ is a one-to-one map ϕ of V into V' such that if $(u, v) \in E$ then $(\phi(u), \phi(v)) \in E'$, for all $(u, v) \in E$. A graph is said to be *cubical* if, for some d, there is an embedding of G into Q_d. If G is cubical, then the least positive number d for which G can be embedded into Q_d is called the *cubical dimension* of G. Another class of embeddings is the isomorphic embedding. An algorithm that determines the isomorphism between G and Q_d in $O(|V|\log_2|V|)$ steps is proposed in [5]. Isomorphism is a very restrictive embedding class since only graphs that have a hypercube topology are isomorphic to Q_d.

Cubical embedding requires only that the adjacencies of G be preserved in Q_d. A graph is said to be *cube-critical* if it is not cubical but all of its proper subgraphs are cubical [16]. For example, the smallest cube-critical graphs are K_3 and $K_{2,3}$. A graph is cubical if and only if it contains no cube-critical subgraphs. Other mapping methods have been suggested to embed cube-critical graphs at the expense of *dilation cost* and *expansion cost.* Dilation cost is the cost incurred due to adjacent nodes in G not being adjacent to Q_d. Expansion cost is a measure of the processor utilization in Q_d and is the ratio of the number of nodes in Q_d to the number of nodes in G. These mapping methods include the class of embeddings where nodes in G are mapped to subcubes rather than nodes of Q_d. A subcube of Q_d can be represented by a vector $(x_1, x_2, \ldots, x_d)$, where $x_i \in \{1, 0, \star\}$, for all $1 \leq i \leq d$, and $\star$ denotes a coordinate value that is either 1 or 0. Given two subcubes $X = (x_1, x_2, \ldots, x_d)$ and $Y = (y_1, y_2, \ldots, y_d)$ the distance between the two subcubes along the ith dimension, $D_i(X,Y)$, is 1 if $\{x_i, y_i\} = \{1, 0\}$; otherwise it is 0. Thus, the *Hamming distance* between X and Y is given by $D(X, Y) = \sum_{i=1}^{d} D_i(X, Y)$. The vectors X and Y are said to be *adjacent* if and only if $D(X, Y) = 1$.

A *squashed-cube embedding,* a concept due to Graham and Pollak [18], is a one-to-one homomorphism from V into a set of mutually disjoint subcubes which preserves the Hamming distance as defined above. The minimum dimension of the hypercube required for a squashed-cube embedding of G is called the *weak cubical dimension* (*WCD*), denoted here as SQ-WCD, of G [11]. Winkler [31], showed that every connected graph, G_n, consisting of n nodes has a distance-preserving squashed-cube embedding in a hypercube Q_{n-1}. In particular, K_n can be squashed-cube embedded into Q_{n-1}.

Chen [11] introduced a new class of embedding called *relaxed squashed-cube* (*RS*) *embedding* which only preserves the *adjacencies* in G. The minimum hypercube dimension required for this embedding will be denoted as RS-WCD. The RS embedding problem is similar to the squashed-cube embedding problem and Chen conjectured that it is NP-complete. RS embedding requires hypercubes of considerably lesser dimension than cubical embedding, and every graph can be RS embedded into a hypercube. However, the method in [11] results in a RS-WCD of $(n - 1)$ for K_n, thus requiring an exponential number of processors to embed K_n.

In studying the problem of state assignment in asynchronous sequential machines, Huffman [22, 23] used Karnaugh maps [24] to show that K_n can be mapped onto a cube, Q_d, of dimension $d = 2\lceil \log_2 n \rceil - 1$, such that each node in K_n is assigned to *two* adjacent subcubes. This is a less restrictive form of RS embedding which we call *relaxed extended squashed* or *RES* embedding and the cube dimension required in this case is called the RES-WCD. For a complete bipartite graph, $K_{m,n}$, this approach provides an RS-WCD of $\lceil \log_2 m \rceil + \lceil \log_2 n \rceil$. Thus, the dimension of the cube required for adjacency-preserving mapping is now $O(\log_2 n)$ rather than $O(n)$, where n is the number of nodes in the graph. Note that E-R schema graphs are, by definition, bipartite.

3. EMBEDDING COMPLETELY CONNECTED GRAPHS

This section discusses the embedding of completely connected graphs and completely connected bigraphs in hypercubes. First, some relevant definitions and terminology are presented.

3.1. *Definitions*

Let σ refer to a string of length $d \geqslant 0$, called a *d-tuple,* where each symbol in the string belongs to the set $\{0, 1, \star\}$. Thus, σ is a member of $\{0, 1, \star\}^d$ and represents a subcube in Q_d.

DEFINITION 3.1. Two distinct d-tuples, σ_1 and σ_2, are said to be *adjacent* iff the Hamming distance between them is 1.

DEFINITION 3.2. Let $\sigma_1, \sigma_2 \in S$, where S is a nonempty set of d-tuples. The pair (σ_1, σ_2) is said to be *connected* if there exists a sequence of adjacent d-tuples of S that starts with σ_1 and ends with σ_2. The set S is said to be a *connected set* if each pair of elements σ_i and $\sigma_j \in S$ is connected.

For example if $S = \{0000\star, 0010\star, 0110\star, 0111\star\}$, $\sigma_1 = 0000\star$, $\sigma_2 = 0111\star$, then σ_1 and σ_2 are connected (via $0010\star$ and $0110\star$) and S is a connected set. Thus, a connected set S containing $|S|$ tuples represents $|S|$ subcubes of Q_d such that each subcube is adjacent to at least one other subcube in S.

DEFINITION 3.3. Two connected sets S_1 and S_2 are said to be *adjacent* if there exists a $\sigma_1 \in S_1$ and a $\sigma_2 \in S_2$ such that σ_1 and σ_2 are adjacent.

DEFINITION 3.4. Let the connected sets $S_1, S_2, \ldots, S_n$ be assigned to nodes $1, 2, 3, \ldots, n$, respectively, of a graph, G_n, of n nodes. This assignment is said to be *adjacency preserving* if, for any two adjacent vertices i, j of G_n, $i \neq j$, S_i is adjacent to S_j.

An n-variable *Karnaugh map* (or K-map) consists of 2^n cells representing all possible combinations of these variables [24]. For example, Fig. 2 illustrates a five-variable K-map. Cyclic code is used to list the combinations as column and row headings. As a result of this coding, cells which have a common side correspond to combinations that differ by the value of just a single variable. Consequently, these cells are adjacent to each other. In general, a cell in an n-variable K-map is adjacent to n different cells. The K-map can be folded along a y-variable in order to determine the adjacent cell of a cell corresponding to that y-variable. For example, in Fig. 2, the cell in the first row, second column is adjacent to the cell in the first row, seventh column along the y_1-variable that corresponds to the first dimension in a Q_5.

To embed K_n into Q_d, every pair of nodes, i and j, in K_n is assigned connected sets, S_i and S_j, in Q_d such that $S_i \cap S_j = \phi$, for all $i \neq j$, and $|S_i| = 2$, $1 \leqslant i, j \leqslant n$, where n is the number of nodes in the graph.

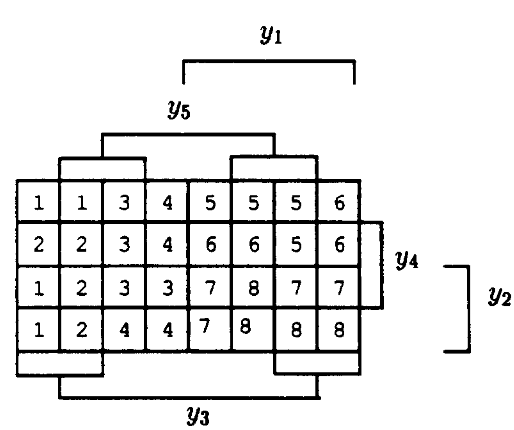


FIG. 2. RES embedding of a K_8 in Q_5 ($d = 2 \log_2 8 - 1$). Nodes in V are labeled 1 through 8.

3.2. *Embedding K_n into Q_d*

The RS-WCD of K_n is $(n - 1)$ using the methods of [11]. Here, we use methods presented in [22, 23] to obtain an RES embedding of K_n which gives an RES-WCD of $2 \times \log_2 n - 1$. This result is obtained by considering a K_n, where $n = 2^k$. First, K_2 can be trivially embedded in Q_1 ($d = 2 \times 1 - 1 = 1$). Next, to embed K_4, at least a cube of dimension 2 is needed. Nodes in Q_2 have 2-bit addresses and Q_2 can be represented by a two-variable K-map. Each cell in the K-map is assigned a distinct node. However, it is not possible to embed K_4 in Q_2 due to the following property.

Property 3.1. Nodes assigned to opposite corners in a two-variable K-map are not adjacent.

Now, consider a K-map of three variables, $y_1y_2y_3$ (i.e., a Q_3). The columns are identified by odd-subscripted variables and the rows by even-subscripted variables. Each node in K_4 is assigned to two adjacent cells, i.e., a Q_1. The left half of the K-map can be represented as rows and the right half as columns. Now, any assignment of the four nodes to the two rows and two columns will result in an RS embedding due to the following.

Property 3.2. Each cell in the row on the left-hand side is adjacent to a cell from a distinct column on the right-hand side.

Thus, for K_4, we have found an embedding in which each node is assigned to one subcube. The result of extending this embedding to K_8 is shown in the five-variable K-map ($d = 2*3 - 1$) of Fig. 2. In general, to embed K_{2^k} we need a Q_d, where $d = 2k - 1$ and each node is mapped to a connected set containing 2^{k-1} nodes. Thus, the following theorem.

THEOREM 3.1. *There exists an RES embedding of K_n into Q_d, where $d = 2k - 1$ and $k = \lceil \log_2 n \rceil$.*

Since the RES-WCD of K_n is logarithmic in n, large graphs can be embedded on reasonable sized hypercubes as compared to previous results. For example, the RS-WCD for K_{32} is 31 according to [11], thus requiring a hypercube of 2^{31} nodes, while the RES-WCD is only 9. Also, the dimensions of the adjacent subcubes assigned by this method are $O(\log_2 n)$ rather than $O(n)$. Finally, since the lower bound on any WCD for K_n is obviously $\lceil \log_2 n \rceil$ and the above theorem now provides an upper bound of $(2 \log_2 n) - 1$, there are $\log_2 n - 1$ choices for the actual dimension.

3.3. *Embedding $K_{m,n}$ into Q_d*

A method for embedding general graphs in hypercubes is discussed in [11]. The graph is decomposed into simple subgraphs and the RS-WCDs of the simple subgraphs are determined. These subgraphs are then recombined using graph operators to once again obtain the original graph. The RS-WCD of the original graph is obtained as the sum of the RS-WCDs of the simple subgraphs plus an additional term which depends on the graph operators employed. An application of this method to complete bigraphs results in an RS-WCD of $\lceil \log m \rceil + \lceil \log n \rceil + 1$ for $K_{m,n}$, as shown. Let $m = |V_1|$ and $n = |V_2|$. For each $w_1 \in V_1$ ($w_2 \in V_2$), assign a $\lceil \log_2 m \rceil$ ($\lceil \log_2 n \rceil$) bits long address, $DV_1(w_1)$ ($DV_2(w_2)$). Assign each node $w \in G_{m,n}$ a subcube address according to the following rules:

1. If $w_1 \in V_1$, then assign w_1 a D bits long address $0{*}{*}{*}\cdots DV_1(w_1)$ which contains $\lceil \log_2 n \rceil$ consecutive $*$'s, and $DV_1(w_1)$ is the $\lceil \log_2 m \rceil$ bits long address.
2. If $w_2 \in V_2$, then assign w_2 a D bits long address $1\,DV_2(w_2){*}{*}{*}\cdots{*}$ which contains $\lceil \log_2 m \rceil$ consecutive $*$'s and $DV_2(w_2)$ is the $\lceil \log_2 n \rceil$ bits long address.

If w_1 is adjacent to w_2 then there exists a subcube, viz., $0DV_2(w_2)DV_1(w_1)$, that is adjacent to the subcube $1\,DV_2(w_2)DV_1(w_1)$ in w_2. For example, consider embedding $K_{4,4}$, which requires a Q_5 ($d = 2 + 2 + 1$). Nodes in V_1 are assigned subcubes $0{*}{*}00$, $0{*}{*}01$, $0{*}{*}10$, and $0{*}{*}11$ and nodes in V_2 are assigned subcubes $100{*}{*}$, $101{*}{*}$, $110{*}{*}$, and $111{*}{*}$.

K-map-based methods to embed $K_{m,n}$ are presented below. First, the result for embedding $K_{m,n}$, when $m + n = 2^k$, is obtained. The generalization to any $K_{m,n}$ follows. Note that the following methods provide RS embeddings and not RES embeddings, as in the case of K_n. Lemmas 3.1 and 3.2 consider two different cases for $K_{m,n}$.

LEMMA 3.1. *A $K_{m,n}$ can be RS embedded into Q_d, where $m = n = 2^{k-1}$, $d = 2k - 2$, and $k > 1$.*

Proof. The proof is by construction. Embedding $K_{m,n}$ when $n = m = 1$, is trivial. For $m > 1$, let $K_{m,n} = ((V_1, V_2), E)$, $|V_1| = |V_2| = 2^{k-1}$, and $\lceil \log_2 m \rceil + \lceil \log_2 n \rceil = 2(k - 1)$. Construct a $(2k - 2)$-variable K-map. Let the y_1y_2-variables designate quadrants of the K-map. Assign one row of quadrant $y_1y_2 = 00$ to each of the first 2^{k-2} nodes of V_1 and assign the remaining 2^{k-2} nodes of V_1 to quadrant $y_1y_2 = 11$ in a similar fashion. Clearly, these nodes have been assigned to subcubes which are adjacent to every column (subcube) in quadrants 01 and 10 (Property 3.2). Each column in quadrants 10 and 01 is assigned to a distinct node of V_2. The adjacency requirements of $K_{m,n}$ are satisfied by this assignment. Each quadrant contains $m/2 = 2^{k-2}$ nodes, thus requiring $(k - 2)$ y-variables. Two quadrants each are required for V_1 and V_2, thus requiring $2(k - 1)$ or $2k - 2$ y-variables for all the nodes in V_1 and V_2. ■

For example, Fig. 3 shows the embedding of $K_{8,8}$ in Q_6 ($d = 2 \times 4 - 2$). Nodes in the node sets, V_1 and V_2, are labeled from 1 to 8 and a to h, respectively.

$\geqslant m$. Thus, $\lceil \log_2(m+n) \rceil \geqslant j_2$. According to Theorem 3.2, the RS-WCD of $K_{m,n}$ is $j_1 + j_2$ and since $G_{m,n}$ is a subgraph of the $K_{m,n}$, the RS-WCD of $G_{m,n}$ is $\leqslant j_1 + j_2$. Furthermore, since any $G_{m,n}$ requires at least a hypercube of dimension $\lceil \log_2(m+n) \rceil$, we have $j_2 \leqslant$ RS-WCD of $G_{m,n} \leqslant j_1 + j_2$.

Consider a graph, $G_{2^{j_1},2^{j_2}} = ((V_1, V_2), E)$, which has the same edge set as $G_{m,n}$. The algorithm for RS embedding $G_{2^{j_1},2^{j_2}}$ is presented. Let C_i, $i \in V_1$, denote the *adjacency* set of a node i; i.e. $C_i = \{k \mid k \in V_2$ and there is an edge between i and $k\}$. Similarly, let C'_j, $j \in V_2$, denote the adjacency set of node j.

Let $V_1^1, V_1^2, \ldots, V_1^{2^{j_1-i+1}}$ ($V_2^1, V_2^2, \ldots, V_2^{2^{j_2-i+1}}$) denote 2^{j_1-i+1} (2^{j_2-i+1}) partitions of the node set V_1 (V_2), for $1 \leqslant i \leqslant j_1$, such that for each partition V_1^k of V_1 the following is true. For every $u, v \in V_1^k$ and $s \ni V_1^k$:

$$|C_u \cap C_v| \geqslant |C_u \cap C_s| \tag{1}$$

and

$$|V_1^k| = 2^{i-1}. \tag{2}$$

Each partition, V_2^k, of V_2 is obtained by similar conditions. For each partition V_1^t, $1 \leqslant t \leqslant 2^{j_1-i+1}$, let U_t denote the set

$$U_t = \{V_2^j \mid (\bigcup_{s \in V_1^t} C_s) \cap V_2^j \neq \phi,\ 1 \leqslant j \leqslant 2^{j_2-i+1}\}.$$

Two partitions V_1^t and V_2^j are connected if there is an edge between nodes u and v in $G_{2^{j_1},2^{j_2}}$, where $u \in V_1^t$ and $v \in V_2^j$. Let $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$ denote a *hypergraph* that contains one node each for every partition of V_2 and $2^{j_2-j_1}$ nodes for every partition of V_1. Let V_1^{tk}, $1 \leqslant k \leqslant 2^{j_2-j_1}$, denote the $2^{j_2-j_1}$ nodes representing each partition V_1^t. Connect V_1^{tk} nodes of each partition V_1^t in a hypercube topology of dimension $j_2 - j_1$ in the hypergraph $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$. Connect each V_1^{tk} to $|U_t|/2^{j_2-j_1}$ distinct $V_2^j \in U_t$ nodes. Based on the topology of the hypergraph, the next lemma states sufficient conditions for embedding a $G_{2^{j_1},2^{j_2}}$ into a Q_d, where $j_2 + 1 \leqslant d \leqslant j_1 + j_2$.

LEMMA 3.3. *If the hypergraph $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$ is isomorphic to Q_{j_2-i+2} then $G_{2^{j_1},2^{j_2}}$ can be RS embedded on a hypercube Q_{j_2+i}.*

Proof. Since $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$ is isomorphic to Q_{j_2-i+2}, there exists an embedding of $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$ such that all the adjacencies in it are satisfied. Note that the graph $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$ contains $2^{j_2-j_1}$ nodes for every partition of V_1 such that if two partitions V_1^t and V_2^j are connected then there is a node corresponding to V_1^t, namely V_1^{tk} in $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$, that is connected to V_2^j. Therefore, all the adjacencies in the partition V_1^t are satisfied. Now, let K_1 be the K-map used to embed $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$ into Q_{j_2-i+2}. Assign a $j_2 - j_1$ dimension subcube each in K_1 to every partition V_1^t of V_1. Assign one node each to every partition V_2^j of V_2. Each cell of K_1 is a K-map of $(2i-2)$ y-variables. Assign one row (column) of a cell to each node in V_1^{tk} (V_2^j). ■

The requirement that the hypergraph be isomorphic to a hypercube is, in fact, too restrictive. In general, it is sufficient that the graph $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$ be *embeddable* in Q_{j_2-i+2}, i.e., that the hypergraph is cubical. However, the problem of deciding whether the graph G_{2^d} is embeddable in Q_d is NP-complete [25]. While it has been reported that Krumme [25] has found an embedding for such graphs into Q_{d+1}, the authors are not aware of the characterization of such graphs. If such an embedding is available, it can be used to determine whether the hypergraph is embeddable in Q_{j_2-i+3}. The following algorithm finds an RS embedding for $G_{2^{j_1},2^{j_2}}$ using the above lemma.

Algorithm Incomplete

(* Embedding an incomplete $G_{2^{j_1},2^{j_2}}$ in a hypercube *)

1. Assign $G = G_{2^{j_1},2^{j_2}} = ((V_1, V_2), E)$; and $\mathcal{G} = \phi$.

(* $\mathcal{G}$ denotes the hypergraph of Lemma 3.3 *)

2. For $i = 1, 2, \ldots, j_1$ do:
   (a) Construct the set C_t for every $t \in V_1, V_2$.
   (b) Generate 2^{j_1-i+1} (2^{j_2-i+1}) partitions of V_1 (V_2) according to (1) and (2).
   (c) Generate U_t for every $t \in V_1$.
   (d) Construct the hypergraph $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$.
   (e) If $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$ is isomorphic to Q_{j_2-i+2} then an RS embedding has been found for $G_{2^{j_1},2^{j_2}}$ on Q_{j_2+i} else $G = \mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}}$; $\mathcal{G}_{2^{j_2-i+1},2^{j_2-i+1}} = \phi$.

Step 2(b) takes $O(n^3)$ time, where n is the number of nodes in G. The graph size reduces by a factor of 2 in each iteration. In the worst case, there are j_1 iterations; i.e., the running time is $O(n^3 j_1)$, where $n = 2^{j_1} + 2^{j_2}$ and $j_2 \geqslant j_1$.

For example, a $G_{16,16}$ is represented in Fig. 5. The algorithm attempts to embed the graph in Q_5 ($d = j_2 + 1 = 5$) and determines that it cannot be embedded. Next, the node sets V_1 and V_2 are each partitioned into eight partitions, $\{\{a, b\}, \{c, d\}, \{e, f\}, \{g, h\}, \{i, j\}, \{k, l\}, \{m, n\}, \{o, p\}\}$ and $\{\{1, 2\}, \{3, 4\}, \{5, 6\}, \{7, 8\}, \{9, 10\}, \{11, 12\}, \{13, 14\}, \{15, 16\}\}$, respectively. The corresponding $\mathcal{G}_{8,8}$ is generated and is represented in Fig. 6. The $\mathcal{G}_{8,8}$ is isomorphic to $Q_{j_2-i+2} = Q_{4-2+2} = Q_4$. Thus, the RS-WCD of the required hypercube is $j_2 + i = 4 + 2 = 6$ (see Fig. 7).

4. DATABASE PROCESSING ISSUES

4.1. *Mapping E-R Schema Graphs*

The Entity–Relationship model views the real world as consisting of entities and relationships. An *entity* is an object which can be distinctly identified and is described by a set of *attributes*. Entities are grouped into *entity sets*, E_i's,

{a : 1, 2, 3, 4, 7, 8, 9, 10}; {b : 1, 2, 3, 4, 7, 8, 9, 10}
{c : 1, 2, 3, 4, 5, 6, 11, 12}; {d : 1, 2, 3, 4, 5, 6, 11, 12}
{e : 3, 4, 5, 6, 7, 8, 15, 16}; {f : 3, 4, 5, 6, 7, 8, 15, 16}
{g : 1, 2, 5, 6, 7, 8, 13, 14}; {h : 1, 2, 5, 6, 7, 8, 13, 14}
{i : 1, 2, 9, 10, 11, 12, 13, 14}; {j : 1, 2, 9, 10, 11, 12, 13, 14}
{k : 3, 4, 9, 10, 11, 12, 15, 16}; {l : 3, 4, 9, 10, 11, 12, 15, 16}
{m : 5, 6, 11, 12, 13, 14, 15, 16}; {n : 5, 6, 11, 12, 13, 14, 15, 16}
{o : 7, 8, 9, 10, 13, 14, 15, 16}; {p : 7, 8, 9, 10, 13, 14, 15, 16}

FIG. 5. An incomplete bipartite graph, $G_{16,16}$. Nodes in V_1 (V_2) are assigned labels a through p (1 through 16). The edges in E are represented by the sets; e.g., {a: 1, 2, 3, 4, 7, 8, 9, 10} indicates that $a \in V_1$ is connected to nodes 1, 2, 3, 4, 7, 8, 9, 10 $\in V_2$.

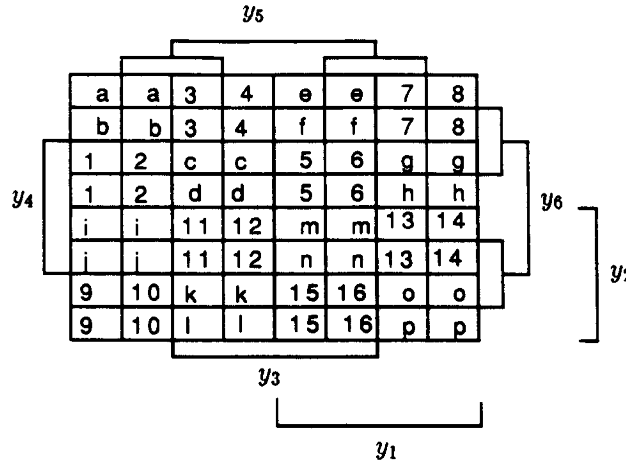


FIG. 7. An RS embedding of the graph of Fig. 5.

such that an entity has to satisfy a predicate in order to be a member of an entity set. A *relationship* is a mathematical relation among n entities such that each entity is taken from an entity set. A set, R_i, of relationships over a given set of entities is called a *relationship set,* where $R_i = \{(e_1, e_2, \ldots, e_n) \mid e_1 \in E_1, \ldots, e_i \in E_i, \ldots, e_n \in E_n\}$.

An E-R schema graph can be mapped into a hypercube by first converting it into a *storage graph,* G, which is an undirected graph containing a node corresponding to each entity and relationship set in the E-R schema. From the definitions, it is clear that entity sets cannot be contained in, or be constituent classes of, other entity sets. Similarly, relationship sets cannot contain other relationship sets. Thus, by definition, the E-R schema graph and its storage graph, G, are bipartite.

Figure 8 shows an E-R schema graph containing the entity sets, Department, Employee, Project, and Supplier, and the relationship sets. Dept–Emp, Proj–Work, Proj–Manager, Supp–Proj, and Proj–Auditor. The meanings of the relationship sets are self-explanatory. For example, Proj–Auditor defines the Project–Auditor relationship between the entities Employee and Project. More than one relationship can be defined on given entity sets; e.g., the relationship sets Proj–Work, Proj–Auditor, and Proj–Manager are all defined on the entity sets Project and Employee. The cardinality of the various relationships between entity sets can be 1:1, 1:n, or m:n. For example, Proj–Auditor is a 1:n relationship between Employee and Project. This implies that a single employee can be the auditor for more than one project and each project has only a single auditor. While the cardinality of relationships and other semantics do not effect embedding, they need to be accounted for by the data storage and processing strategies. Figure 9 shows the corresponding storage graph for the schema. The graph for this example is a $G_{4,5}$ that contains a cube-critical graph, $K_{2,3}$. The lower bound on the cube dimension for this graph is 4. The embedding algorithm starts with $G_{4,8}$ and embeds it in Q_5, as shown in Fig. 10. This example illustrates several fea-

{{a, b} : {1, 2}, {3, 4}, {7, 8}, {9, 10}}
{{c, d} : {1, 2}, {3, 4}, {5, 6}, {11, 12}}
{{e, f} : {3, 4}, {5, 6}, {7, 8}, {15, 16}}
{{g, h} : {1, 2}, {5, 6}, {7, 8}, {13, 14}}
{{i, j} : {1, 2}, {9, 10}, {11, 12}, {13, 14}}
{{k, l} : {3, 4}, {9, 10}, {11, 12}, {15, 16}}
{{m, n} : {5, 6}, {11, 12}, {13, 14}, {15, 16}}
{{o, p} : {7, 8}, {9, 10}, {13, 14}, {15, 16}}

FIG. 6. The hypergraph, $\mathcal{G}_{8,8}$, of the graph in Fig. 5 for the second iteration of the algorithm Incomplete in Section 3.4. {x, y} represents a partition of the nodes in V_1 or V_2. The edges of $\mathcal{G}_{8,8}$ are represented by sets; e.g., {{a, b}: {1, 2}, {3, 4}, {7, 8}, {9, 10}} indicates that the partition {a, b} is connected to the partitions {1, 2}, {3, 4}, {7, 8}, {9, 10}.

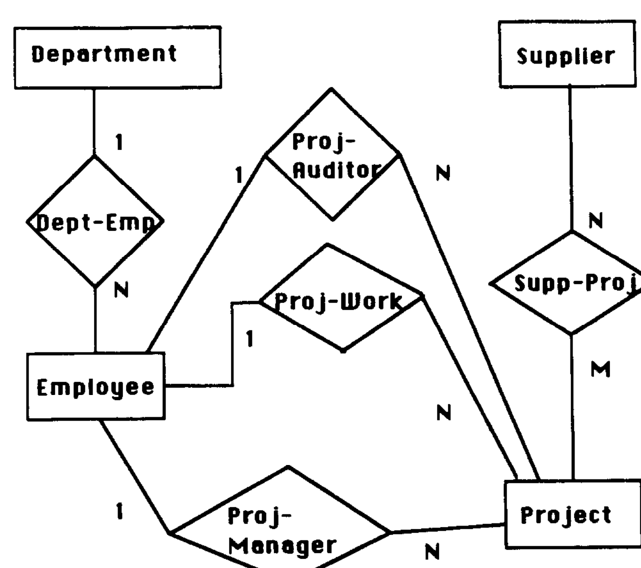


FIG. 8. A sample E-R logical schema graph.

tures of the RS embedding for E-R schemas which are discussed in the next section.

4.2. *Ramifications of RS Embedding*

We began studying the problem of embedding logical schema graphs into processor graphs in order to solve the data placement problem for parallel database systems. The intent was to find an embedding scheme that minimizes communication costs in a multicomputer system. Squashed embedding is preferred because (1) it distributes data across subcubes, thereby providing parallel I/O capability. (2) Queries can be represented as subgraphs of the schema graph. Only the dataflow directions in the query graph need to be determined at run time, based on the selection predicates in the query. We are currently studying various aspects of this problem. (3) It should be possible to include the dataflow concurrency control methods that have been proposed in the literature [14] into this dataflow processing scheme. (4) The reliability of the database system could be enhanced in several ways. Failure of a node and/or link implies that the corresponding entity or relationship node is no longer accessible in the logical schema graph. Thus, future queries could be classified as being fully answerable, partially answerable, or unanswerable. (5) Once a schema graph is mapped, one or more replicas of the entire database can be easily mapped onto the same hypercube by "skewing" each copy, by applying a simple address transformation at each node. Thus, each node in the hypercube knows the exact location of its copies.

However, direct embedding of logical schema graphs into processor graphs, using adjacency-preserving techniques, has several other ramifications for database processing. First, there is little control over the size of the subcube assigned to each node in the schema graph. Thus, it is not possible to choose subcube sizes in proportion to, say, the amount of data represented by each node. Second, the embedding may result in unused subcubes, as in Fig. 10. In general, squashed embedding may have to be used along with other schemes to obtain a good mapping. The above issues are all topics for future research.

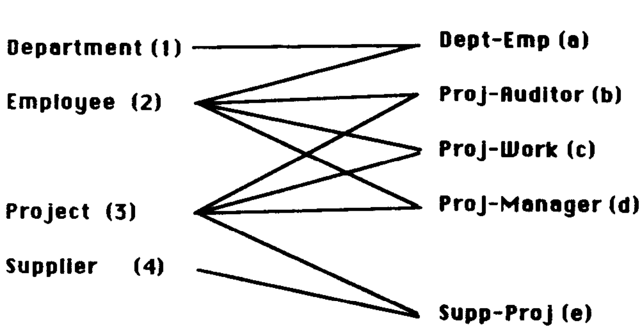
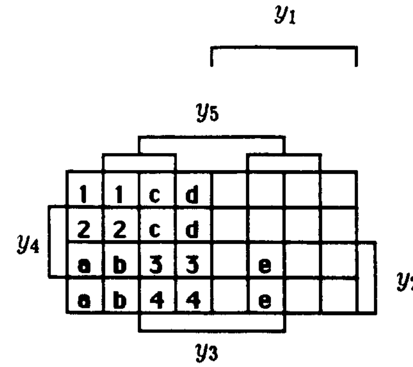


FIG. 9. The $G_{4,5}$ storage graph corresponding to the E-R schema graph of Fig. 8.



1 = (0000*); 2 = (0001*)
3 = (01111); 4 = (0110*)
a = (010*0); b = (010*1)
c = (001*1); d = (001*0)
e = (111*1)

FIG. 10. An RS embedding of the $G_{4,5}$ of Fig. 9 into a Q_5.

5. CONCLUSION

A class of adjacency-preserving, squashed embeddings was studied with the idea of using such embeddings to directly map E-R schema graphs onto hypercubes. Results for relaxed extended squashed embedding of complete graphs and relaxed squashed embedding of complete bigraphs were obtained using a technique that was used in the state assignment of asynchronous sequential machines by Huffman. The minimum cube dimension required for such embedding is called the weak cubical dimension of the graph. For K_n, an RES-WCD of $O(\log_2 n)$ is obtained. For $K_{m,n}$, the RS-WCD is $\lceil \log_2 m \rceil + \lceil \log_2 n \rceil$. Thus, the bounds on the RS-WCD of $G_{m,n}$ are $d_{min} = \lceil \log_2(m+n) \rceil \leq$ RS-WCD $\leq \lceil \log_2 m \rceil + \lceil \log_2 n \rceil = d_{max}$. An algorithm for determining the RS-WCD of incomplete bipartite graphs was presented which examines the $d_{max} - d_{min}$ choices to be examined to obtain the best cube dimension. However, while preserving adjacency is important when mapping schema graphs, another important factor is the loading of each node/subcube. In general, algorithms for embedding should consider this factor also. Overall, this embedding approach provides several opportunities for novel query processing, concurrency control, and reliability methods in parallel database systems.

REFERENCES

1. Abiteboul, S., and Hull, R. IFO: A formal semantic database model. *ACM Trans. Database Systems* **12,** 4 (Dec. 1987), 525–565.
2. Banerjee, J., Hsaio, D. K., and Kannan, K. DBC: A database computer for very large databases. *IEEE Trans. Comput.* **C-28,** 6 (June 1979).
3. Baru, C. K., and Frieder, O. Database operations in a cube-connected multicomputer system. *IEEE Trans. Comput.* **C-38,** 6 (June 1989), 920–927.

4. Baru, C. K., Padmanabhan, S., Stout, Q., and Wagar, B. A comparison of join and projection algorithms for hypercubes. *Proc. 4th Conference on Hypercube Concurrent Computers and Applications*, Monterey, CA, Mar. 6–8, 1989.
5. Bhat, K. V. S. On complexity of testing a graph for n-cube. *Inform. Process. Lett.* **11**, 1 (Aug. 1980), 16–19.
6. Bic, L., and Hartmann, R. L. Simulated performance of a data-driven database machine. *J. Parallel Dist. Comput.* **3** (1986), 1–22.
7. Bic, L., and Rasset, T. L. Performance of a relational dataflow machine. *Proc. Hawaii International Conference on System Sciences*, Jan. 1986, pp. 1–22.
8. Bic, L., and Hartmann, R. L. AGM: A dataflow machine. Tech. Rep. Department of Information and Computer Science, University of California, Irvine, Feb. 1987.
9. Bratbergsengen, K. Algebra operations on a parallel computer: Performance evaluation. *Proc. 5th International Workshop on Database Machines*, Karuizawa, Japan, Oct. 1987.
10. Chen, P. P. S. The entity-relationship model: Toward a unified view of data. *ACM Trans. Database Systems* **1**, 1 (Mar. 1976), 9–36.
11. Chen, M. S., and Shin, K. G. Embedding cooperative task modules into a hypercube multiprocessor. *Proc. 2nd Conference on Hyp. Conc. Computers and Applications*, Monterey, CA, 1987, pp. 122–129.
12. DeWitt, D. J. DIRECT: A multiprocessor organization for supporting relational Database Management Systems. *IEEE Trans. Comput.* **C-28**, 6 (June 1979), 395–408.
13. DeWitt, D. J., *et al.* GAMMA: A high performance dataflow database machine. *Proc. 12th International Conference on Very Large Data Bases*, Kyoto, Japan, Aug. 1986, pp. 228–237.
14. Eich, M. H., and Wells, D. L. Database concurrency control using data flow graphs. *ACM Trans. Database Systems* **13**, 2 (June 1988), 197–227.
15. Elmasri, R., Weeldreyer, J., and Hevner, A. The category concept: An extension to the entity-relationship model. *Data and Knowledge Engineering*, Elsevier/North-Holland, New York, 1985, Vol. 1.
16. Garey, M. R., and Graham, R. L. On cubical graphs. *J. Combin. Theory Ser. B* **18** (1975), 84–95.
17. Graham, R. L., and Pollak, H. O. On the addressing problem for loop switching. *Bell System Tech. J.* **50**, 8 (Oct. 1971), 2495–2519.
18. Graham, R. L., and Pollak, H. O. On embedding graphs in squashed cubes. *Graph Theory and Application*, Lecture Notes in Mathematics Vol. 303. Springer-Verlag, New York/Berlin (Proc. of a conference held at Western Michigan University, May 10–13, 1972).
19. Hammer, M. M., and McLeod, D. J. Database description with SDM: A semantic database model. *ACM Trans. Database Systems* **6**, 3 (Sept. 1981).
20. Harary, F., Hayes, J. P., and Wu, H. J. A survey of the theory of hypercube graphs. *Comput. Math. Appl.* **15**, 4 (1988), 277–289.

Received February 15, 1989; revised September 28, 1989

21. Hillyer, B., and Shaw, D. E. NON-VON's performance on certain database benchmarks. *IEEE Trans. Software Engrg.* **SE-12**, 4 (Apr. 1986), 577–583.
22. Huffman, D. A. The synthesis of sequential switching circuits: Part I. *J. Franklin Inst.* **257** (Mar. 1954), 161–190.
23. Huffman, D. A. The synthesis of sequential switching circuits: Part II. *J. Franklin Inst.* **257** (Apr. 1954), 275–303.
24. Kohavi, Z. *Switching and Finite Automata.* McGraw-Hill, New York, 1978.
25. Livingston, M., and Stout, Q. F. Embeddings in hypercubes. *Math. Comput. Modelling* **11** (1988), 222–227.
26. Padmanabhan, S., and Baru, C. K. Single query execution in multicomputer database systems. Tech. Rep., Adv. Comp. Arch. Lab., University of Michigan, Ann Arbor, MI.
27. Su, S. Y. W. SAM*: A semantic association model for corporate and scientific-statistical databases. *Inform. Sci.* **29** (1983), 151–199.
28. Su, S. Y. W. *Database Computers.* McGraw-Hill, New York, 1988.
29. Teorey, T. J., Yang, D., and Fry, J. P. A logical design methodology for relational databases using the extended entity-relationship model. *ACM Comput. Surveys* **18**, 2 (June 1986), 197–222.
30. Verheijen, G. M. A. NIAM—An information analysis method. Information Systems Department, Control Data B V, The Netherlands, 1976.
31. Winkler, P. M. Proof of the squashed cube conjecture. *Combinatorica* **3**, 1 (1983), 135–139.

CHAITANYA K. BARU received the bachelor's degree (B.Tech.) from the Indian Institute of Technology, Madras, India, in 1979 and the master's (M.E.) and Ph.D. degrees from the University of Florida, Gainesville, Florida, in 1983 and 1985, all in electrical engineering. He has been an assistant professor in the Department of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor, Michigan, since 1985. His research interests are in the areas of database systems, parallel processing, and database machines. One of his current projects is the implementation of an experimental database system on a transputer-based parallel computer. His research is currently funded by the National Science Foundation.

PIYUSH GOEL received a bachelor's degree in electronics and communication engineering from the University of Roorkee, Roorkee, India, in 1982 and a master's degree in electrical and computer engineering from Wayne State University, Detroit, Michigan, in 1984. He is currently pursuing a Ph.D. degree in the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, Michigan. His current research is related to parallel processing of databases based on semantic data models and query execution strategies and concurrency control issues in such systems.